ENGN 4528 Computer Vision, Semester 1, 2003

Lab 1: Introduction to Image Processing in Matlab & Binary Image Analysis

Welcome to your first Computer Vision Lab. This lab will introduce you to writing image processing software using Matlab and the Matlab Image Processing Toolbox. You will learn how to load, view and manipulate images, convert between image formats, threshold images, and you will write a function that calculates the moments and orientation of a binary object.

Deliverable:

To gain the marks for this lab you will need to show me your `Lab1` function running during the lab. The specification for this function is towards the end of this document.

To get started quickly work through the Getting Started and Introduction to Image Processing in Matlab sections.

Getting Started

1. Make a new directory in which to do your work for this course.
2. Download the image 'text.tiff' from
   http://www.syseng.anu.edu.au/~luke/cvcourse_files/images/text.tiff
   Save this image in your new directory.
3. Open Matlab from the Start>Programs menu.
4. Change to your new directory (use `cd`).
5. Type `edit` to open a new script file, and you are ready to start work.

I suggest you do all your work in your script file.

Did you know:

- You can write multiple Matlab functions in the same file? This is helpful to keep your work together especially when working on large projects, to do this you will have to make the file a function.
- You can use the debugging features in the Matlab editor to jump between the local variable spaces when debugging your functions. Check 'Stop if Error' in the Breakpoints menu of the Matlab editor to activate the debugger on the next error. Note the different cursor in debugging mode.
- `dbquit` exits debugging mode. Remember to quit debugging mode before rerunning your code.

Introduction to Image Processing in Matlab

You can get help on any Matlab function by typing `help <function>` or to get help specifically on the Image Processing Toolbox type `help images`, typing just `help` will show you all the available help topics.

Load the image into im_rgb with

```
im_rgb = imread('text.tiff');
```

convert to type double

```
im_rgb = double(im_rgb);
```

convert to grey-scale

```
im_grey = (im_rgb(:,:,1)+im_rgb(:,:,2)+im_rgb(:,:,3))/3;
```

or you can use `rgb2gray` which gives a slightly different result since it converts to a *luminance* rather than an *intensity* image, you'll learn about this in the lecture on colour imaging. For this lab it doesn't matter which one you use.

convert from [0,255] to [0,1]

```
im_grey = im_grey/255;
```

to view an image use either `imshow(im)` or `imagesc(im)`, note the difference. You can use `imagesc` and then set the axis for an image with

```
axis image;
```

Axis labelling can be turned on and off by

```
axis on;
axis off;
```

You can interactively crop an image using `imcrop(im_grey);` Try this. For this lab exercise I would like you to crop the image to a specified region containing only the 2nd letter. Do this with

```
im_grey = im_grey(150:270, 280:400);
```

Examine the intensity profile of im_grey

```
improfile(im_grey);
```

Threshold im_grey

```
im_bin = imgrey>0.5;
```

The object we are interest in is the text, since we have defined binary objects to be 1's and the background to be zero we need to invert the image

```
im_bin = 1 – im_bin;      % inverts the binary image.
```

You can find the coordinates of the points in this object using `find`. Type `help find` to learn about this function. Write a line of code using `find` that returns the x and y values of all the points in the object.

Now you're ready to get to work on the assessable task.

Specification for Lab 1 Deliverable:

Write a Matlab function `Lab1.m` with a sub-function `find_moments`. This means your file will be called `Lab1.m` and shall contain two functions `Lab1` and `find_moments`. In future you will find it easier to manage larger projects if you keep all the code for each project in a single file.

The specifications of these functions follow.

```
function Lab1;
% This function –
% - loads an image,
% - displays the image and its intensity histogram,
% - converts the image to binary via a threshold
% - calculates the area, centre of mass and orientation of
%       the binary object.
% - displays the binary image with its centre of mass
%       indicated and a line showing its orientation.
% - prints out the area, centre of mass, and orientation
```

```
function [moments, orientation]  = find_moments(I_bin);
% Calculates the 0th, 1st and 2nd moments and orientation of
% the binary image I_bin and
% returns:
%   moment.M = 0th moment (area)
%   moment.Mx = 1st moment (x-coordinate of centroid)
%   moment.My = 1st moment (y-coordinate of centroid)
%   orientation = orientation
```

When writing `find_moments` you will find it useful to refer to the lecture notes on binary moments. You can access these online from www.syseng.anu.edu.au/~luke/cvcourse.htm.

You might like to try your code on other letters in text.tiff as well, or to construct some artificial data to test your system.

Additional Exercises (not directly assessable during the lab)

Load in some other images, you can get images off the net, from the image folder on the course website, or there are some images that come with the Image Processing Toolbox that you can load from any directory whilst in Matlab these images include:

- saturn.tif (note '.tif' not '.tiff')
- pout.tif, bonemarr.tif
- eight.tif, tire.tif

Experiment with

```
impixel, improfile, histeq, colormap, bwmorph, bwlabel,
```

Type help <function> to see the specs for the functions then see if you can implement them. I suggest trying the following quick exercises:

- Use `bwlabel` to label a point in a binary image
- Experiment with dilation, erosion, opening and closing using `bwmorph`, try and simulate the results presented in lectures.